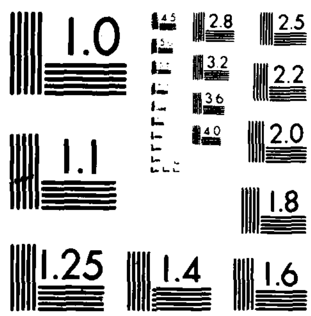
AD-A086 564 NAVAL POSTGRADUATE SCHOOL MONTEREY CA F/G 15/7
SOME UTILITY PROGRAMS FOR THE STAR COMBAT SIMULATION MODEL.(U)
JUN 80 J K HARTMAN MIPR-CD-1-80
UNCLASSIFIED NPS55-80-022 NL

1 OF 1
AD
A086564

END
DATE
FILMED
8-80
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

LEVEL

NPS55-80-022

NAVAL POSTGRADUATE SCHOOL

Monterey, California

JUL 1 6 1980

A

SOME UTILITY PROGRAMS FOR THE
STAR SIMULATION MODEL

by

James K. HARTMAN

June 1980

Approved for public release; distribution unlimited.

Prepared for:
The U.S. Army Training & Doctrine Command
Fort Monroe, VA.

NAVAL POSTGRADUATE SCHOOL
Monterey, California

Rear Admiral J. J. Ekelund
Superintendent

Jack R. Borsting
Provost

The work reported herein was accomplished with the support of the U.S. Army Training & Doctrine Command, Fort Monroe, VA.

Prepared by:

James K Hartman

James K. Hartman
Department of Operations Research

Reviewed by:

G Howard Acting Chm

Michael G. Sovereign, Chairman
Department of Operations Research

Released by:

William M. Tolles

William M. Tolles
Dean of Research

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER 14 NPS55-80-022	2. GOVT ACCESSION NO. AD-A086 564	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) 6 Some Utility Programs for the STAR Combat Simulation Model		5. TYPE OF REPORT & PERIOD COVERED 9 Technical Report.
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) 10 James K. Hartman		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 15 MIPR-CD-1-80
11. CONTROLLING OFFICE NAME AND ADDRESS U.S. Army TRADOC Fort Monroe, VA 23651		12. REPORT DATE 11 June 1980
		13. NUMBER OF PAGES 22
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) 12 26		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

STAR
Combat models
Contour map
Plotter

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report is a user's manual for several utility programs which have been written to aid the program development effort for the STAR combined arms combat simulation model.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

251450

TABLE OF CONTENTS

I. INTRODUCTION

This report provides a user's manual for several utility programs, which have been used as aids during the development of the STAR combat simulation model.

Section II describes a program for rapidly producing contour maps of the STAR battlefield area and overlaying information of various types on the maps.

Section III describes a program for reconciling two computer programs which are mostly identical, but contain some differences. Such programs have frequently resulted during the STAR development as several groups simultaneously build new modules for incorporation into the base program.

Section IV describes a program for producing Tech report or thesis - ready copies of computer code without requiring photo-reduction.

II. BATTLEFIELD MAP PLOT PROGRAMS

As an aid to visualizing the progress of a simulated battle, and especially as an aid in the scenario formulation process, a program has been written to rapidly prepare contour maps of the STAR battlefield area with a series of optional overlays. The program is written in FORTRAN and uses standard VERSATEC plotter routines.[ref. 1] Several data sets defining the maps are stored on disk by preprocessor programs and read back by the plot program. This saves substantial time since the preprocessing is quite lengthy for preparing the contour map. Section A of this chapter will describe the plot program options. Section B briefly discusses the preprocessor programs.

A. Plot Options

A program listing for the map plot program is given in Figure 1. This particular program is set up to plot a 10 x 10 kilometer map segment at a scale of 1:25000 yielding a map 40 x 40 cm. (another version of the program yields a 30 x 10 km. map area). Plot units for the program are meters, so map coordinates to be plotted are in the range 0. to 10,000. for both X and Y coordinates. The program currently has 9 active options which are selected by the first input data card: a 1 in column j selects option j and a 0 in column j suppresses the option (j=1,...,9).

1. Option 1 Coordinate Grid. This option, if selected, plots a 1 x 1 km grid over the map area.

2. Option 2 Label Coordinates. This option prints UTM coordinates for each kilometer in the map boundary.

3. Option 3 Terrain Contour Map. Option 3 plots a contour map of the battlefield area. Substantial preprocessing (see section B) is done to prepare and place on a disk file the X, Y coordinates defining each contour

```
REPLACE THIS CARD WITH STANDARD JOB CARD WITH TIME = 5
// EXEC FORTCLGW
//FORT.SYSIN DD *
C PLOT PROGRAM FOR 10 X 10 KM TERRAIN BOX.
         DIMENSION IOPT(10),BX(7),BY(7),EX(7),EY(7),X(3600),Y(3600),
        1  ITR(100),ITITLE(20)
         DATA BX/-500.,10500.,10500.,-500.,-500.,    0.,1./,
        1      BY/-500.,-500.,10500.,10500.,-500.,    0.,1./,
        2      EX/0.,10000.,10000.,0.,0.,0.,1./,
        3      EY/0.,0.,10000.,10000.,0.,0.,1./
         DATA LMASK1/Z0F0F/
C
C INPUT OPTIONS AS  1 = DESIRED,  0 = NOT DESIRED
C        IOPT(1) -- COORDINATE GRID
C             (2) -- LABEL COORDINATES
C             (3) -- TERRAIN CONTOUR MAP
C             (4) -- ACCENT CONTOURS DIVISIBLE BY 100.
C             (5) -- FORESTS SHADED
C             (6) -- DRAW LINES (EG. ROUTES)
C             (7) -- DRAW SYMBOLS (EG. POSITIONS)
C             (8) -- TITLE
C             (9) -- OUTLINE ELLIPSES (EG. FIELDS)
C
         READ(5,7) IOPT
 7       FORMAT(10I1)
C BATTLEFIELD LOWER LEFT CORNER COORDS IN METERS
         XLOBY=50000.
         YLOBY=93000.
C
C PLOT FRAME
C
         CALL PLOTS(0,0,0)
         CALL NEWPEN(5)
         CALL LINE(BX,BY,5,1,0,0)
         CALL LINE(EX,EY,5,1,0,0)
         CALL NEWPEN(1)
C
C PLOT COORDINATE GRID
C
         IF (IOPT(1).NE.1) GO TO 200
         WRITE(6,107)
 107     FORMAT(' OPTION 1 -- COORDINATE GRID')
         CALL GRID(0.,0.,10,1000.,10,1000.,LMASK1)
C
C PLOT COORDINATE LABEL NUMBERS
C
 200     IF(IOPT(2).NE.1) GO TO 300
         WRITE(6,207)
 207     FORMAT(' OPTION 2 -- COORDINATE LABELS')
         CX=-100.
```

Figure 1. Map Plot Program

```
      CY=-315.
      CYT=10185.
      DX=-445.
      DXT=10055.
      DY=-65.
      XB=XLOBY/1000.
      YB=YLOBY/1000.
      HT = 130.
      DO 250 I=1,11
      CALL NUMBER(CX,CY,HT,XB,0.0,-1)
      CALL NUMBER(CX,CYT,HT,XB,0.0,-1)
      CX=CX+1000.
      XB=XB+1
      CALL NUMBER(DX,DY,HT,YB,0.0,-1)
      CALL NUMBER(DXT,DY,HT,YB,0.0,-1)
      DY=DY+1000.
  250 YB=YB+1
C
C PLOT TERRAIN CONTOUR MAP
C
  300 IF(IOPT(3).NE.1) GO TO 500
      WRITE(6,307)
  307 FORMAT(' OPTION 3 -- TERRAIN CONTOUR LINES')
      IF (IOPT(4).EQ.1) WRITE(6,407)
  407 FORMAT(' OPTION 4 -- ACCENT 100 M. CONTOURS')
  310 READ(3,317,END=390) NP,CV
  317 FORMAT(I5,F10.0)
      READ(3,327) (X(I),Y(I),I=1,NP)
  327 FORMAT(8F10.2)
      X(NP+1)=0.
      X(NP+2)=1.
      Y(NP+1)=0.
      Y(NP+2)=1.
      CALL NEWPEN(1)
      IF(IOPT(4).NE.1) GO TO 350
C
C     ACCENT CONTOURS DIVISIBLE BY 100.
C
      ICV=CV/100.
      XXCV=CV-ICV*100.
      IF(ABS(XXCV).LT.0.1)  CALL NEWPEN(4)
  350 CALL LINE(Y,X,NP,1,0,0)
      GO TO 310
C OUT OF DATA
  390 CALL NEWPEN(1)
C
C SHADE FORESTED AREAS
C
  500 IF(IOPT(5).NE.1) GO TO 600
      WRITE(6,517)
```

Figure 1 (Continued).

```
517   FORMAT(' OPTION 5 -- SHADE FORESTS')
      HT=75.
      XC=50.
      DO 570 I=1,100
      READ(2,507) (ITR(J),J=1,100)
507   FORMAT(50I1,30X)
      YC=50.
      DO 540 J=1,100
      IF(ITR(J).EQ.0) GO TO 520
      CALL SYMBOL(XC,YC,HT, 9,0.,-1)
520   YC=YC+100.
540   CONTINUE
      XC=XC+100.
570   CONTINUE
C
C PLOT LINES (EG. ROUTES)
C
600   IF(IOPT(6).NE.1) GO TO 700
      WRITE(6,627)
627   FORMAT(' OPTION 6 -- PLOT LINES')
      CALL NEWPEN(2)
610   READ(5,607) NP
607   FORMAT(I5)
      IF(NP.EQ.999) GO TO 690
      WRITE(6,607) NP
      READ(5,617) (X(I),Y(I),I=1,NP)
      WRITE(6,617) (X(I),Y(I),I=1,NP)
617   FORMAT(8F10.0)
      DO 640 I=1,NP
      X(I) = X(I) -XLOBY
640   Y(I) = Y(I) - YLOBY
      X(NP+1)=0.
      X(NP+2)=1.
      Y(NP+1)=0.
      Y(NP+2)=1.
      CALL LINE(X,Y,NP,1,0,0)
      GO TO 610
690   CALL NEWPEN(1)
C
C PLOT SYMBOLS (EG. POSITIONS)
C
700   IF(IOPT(7).NE.1) GO TO 800
      WRITE(6,707)
707   FORMAT(' OPTION 7 -- PLOT POSITIONS')
      HT = 50.
710   READ(5,717) XC,YC,ISYM
717   FORMAT(2F10.0,I5)
      IF(ISYM.EQ.999) GO TO 800
      XC=XC-XLOBY
      YC=YC-YLOBY
```

Figure 1 (Continued).

```
      CALL SYMBOL(XC,YC,HT, ISYM,0.,-1)
      GO TO 710
C
C PLOT TITLE
C
800   IF(IOPT(8).NE.1) GO TO 900
      READ (5,807) ITITLE
807   FORMAT(20A4)
      WRITE(6,817) ITITLE
817   FORMAT(' OPTION 8 -- TITLE  ',20A4)
      XC=-700.
      YC=-400.
      HT=135.
      CALL SYMBOL(XC,YC,HT,ITITLE,90.0,80)
C
C OUTLINE ELLIPSES
C
900   IF(IOPT(9) .NE.1) GO TO 1000
      WRITE(6,907)
907   FORMAT(' OPTION 9 -- PLOT ELLIPSES')
910   READ(5,917) NUM,XC,YC,SAMAJ,SAMIN,ANGLE
      WRITE(6,917) NUM,XC,YC,SAMAJ,SAMIN,ANGLE
917   FORMAT(I5,5F10.2)
      IF(NUM .EQ. 999) GO TO 1000
      CALL NEWPEN(1)
      XC=XC-XLOBY
      YC=YC-YLOBY
      TWOPI=3.14159265*2.0
      ANGLE=ANGLE * TWOPI / 360.
      ASQ= SAMAJ**2
      BSQ=SAMIN**2
      C=TWOPI * SQRT((ASQ+BSQ)/2.0)
      ZN=C/50.0
      N=ZN+1
      NC=-1
      THETA=0.0
      DO 950 I=1,N
      RSQ=(ASQ*BSQ)/(ASQ*(SIN(THETA)**2)+BSQ*(COS(THETA)**2))
      R=SQRT(RSQ)
      APT=ANGLE + THETA
      XX=XC+R*COS(APT)
      YY=YC+R*SIN(APT)
      CALL SYMBOL(XX,YY,25.,1,0.,NC)
      NC=-2
      THETA=THETA + 50.0/R
950   CONTINUE
      GO TO 910
1000  CONTINUE
      CALL PLOT(0.,0.,999)
      STOP
```

Figure 1 (Continued).

```
         END
REPLACE THIS CARD WITH A STANDARD ORANGE END OF FILE CARD
//GO.PLOTPARM DD *
 &PLOT XMIN=-999.,XMAX=12000.,YMIN=-999.,YMAX=12000.,UNITS=.0254,
 SCALE=.00004,STRIP=14000.   &END
//GO.FT02F001 DD UNIT=2314,VOL=SER=PAT001,DSN=PLTFLTR,DISP=SHR
//GO.FT03F001 DD UNIT=2314,VOL=SER=PAT001,DSN=PLTFL50,DISP=SHR
//GO.SYSIN DD *
1111111110
    3
52300.    96150.    51620.    96350.    50950.    93100.
  999
52350.    96200.       14
52400.    96200.       00
52450.    96300.       14
0.0       0.0         999
TEST PLOT FOR TECH REPORT
    1 51500.   95000.    500.      200.      45.
  999
REPLACE THIS CARD WITH A STANDARD ORANGE END OF FILE CARD
```

Figure 1 (Continued).

line on the map. This file, which contains perhaps 50,000 X, Y pairs is read by the plot program and plotted. The current file uses 10 m. contour spacing.

4. Option 4. Accent Contours. If this option is selected along with option 3, then contour lines corresponding to elevations evenly divisible by 100 will be plotted darker than the others.

5. Option 5. Shade Forests. The plot program reads from a disk file a 100m. grid of 0-1 values over the entire battlefield. When a 1 is read, signifying the presence of forests in that 100 x 100 grid, a Y symbol is plotted on the map.

6. Option 6. Draw Lines. If option 6 is selected, the plot program will read X, Y coordinates from cards, and connect them by straight lines. This option is useful for plotting routes for moving vehicles or firer/target pairs for each shot. The data should be punched as follows:

Card 1 NP - number of points to follow (Format I5)

Card 2 X_1 Y_1 X_2 Y_2 etc. for NP points (Format 8F10.0)

⋮ continuing on successive cards as needed.

Card k NP - for the next line to be plotted

Card k+1 X, Y, X, Y coordinates for the line

⋮

Card n NP = 999 signal to end option 6.

7. Option 7 Plot Symbols. This option reads data from cards and plots the indicated symbols on the map. The option is useful for plotting snapshots of element positions during the simulated battle. One card is read for each symbol to be plotted. The card contains three values: X, Y, (the X and Y coordinates of the symbol location on the map) and ISYM (an integer code for the symbol to plot) read in format (2F10.0, I5). The list of available

symbol codes is given in the VERSATEC Graphics Manual, reference [1], page B-4. A symbol value ISYM = 999 terminates the option.

8. Option 8 Print Title. Option 8 reads a single card of character data and plots it on the left hand border of the map.

9. Option 9 Plot Ellipses. Option 9 reads from cards the parameters of ellipses which are then plotted overlaying the battlefield. This option is useful for showing the location of minefields and other areas on the battlefield. One card is read for each ellipse containing the following values:

NUM	-	sequence number - (999 for end of data)	(I5)
XC	-	X coordinate of ellipse center	(F10.2)
YC	-	Y coordinate of ellipse center	(F10.2)
SAMAJ	-	length of semi-major axis	(F10.2)
SAMIN	-	length of semi-minor axis	(F10.2)
ANGLE	-	angle in degrees measured counterclockwise from east to the major axis	(F10.2)

A section from a map plotted by this program is included as Figure 2. All of the options were used in its creation. The data input cards for this plot are shown in Figure 1.

B. Preprocessor Programs and Disk Data Files

The data used to plot the contour lines and forested areas is computed in advance and stored on disk to speed execution of the plot program.

The forest preprocessor does a simple 100m. scan of the battlefield area and evaluates tree height at each sample point. Tree height of zero is stored as a 0, and does not plot a symbol at that point. Positive tree height

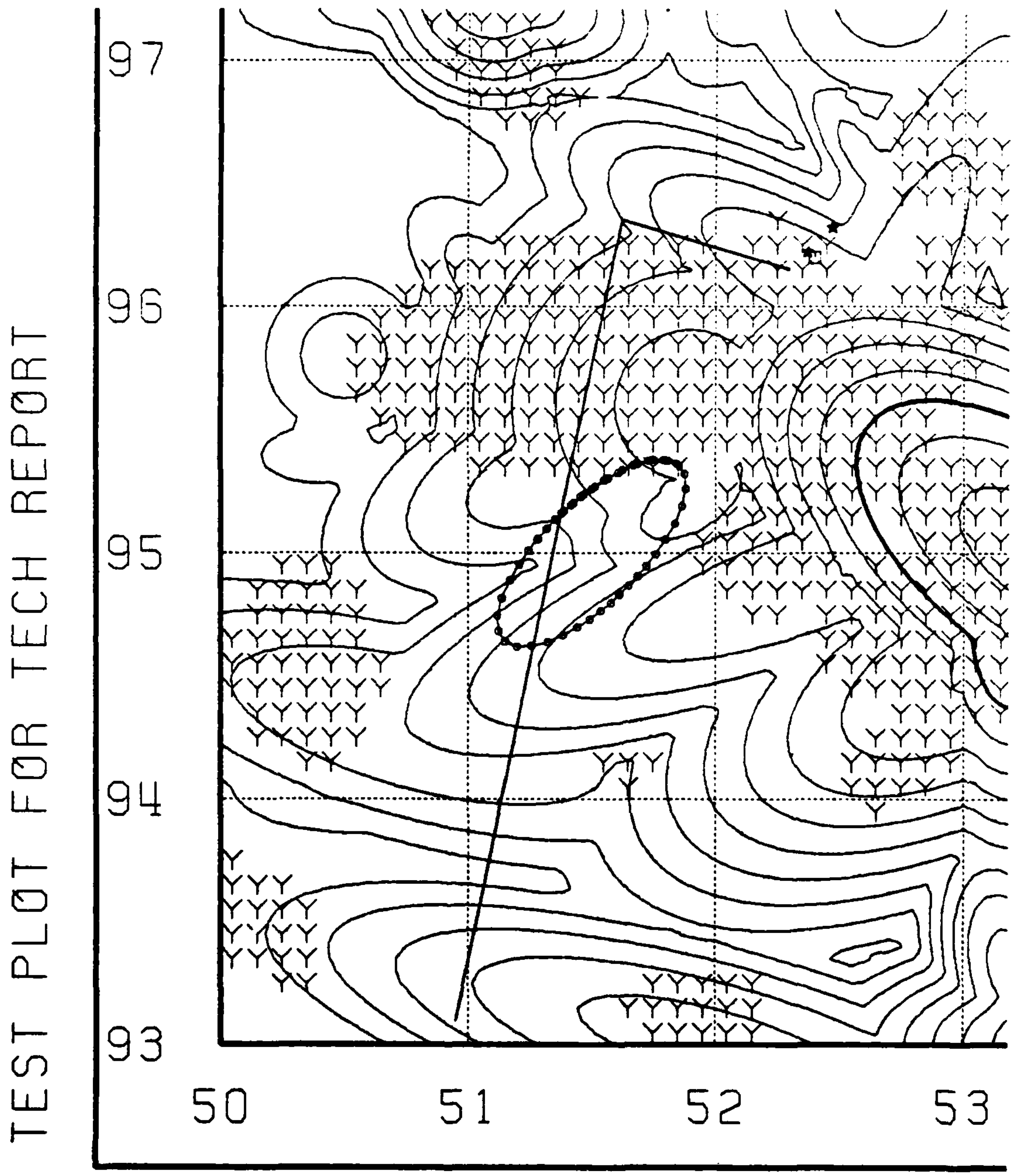


Figure 2. Sample Map

results in a 1 stored on disk and a Y symbol plotted. The tree data is read from FORTRAN data set FT02F001 by the plot program.

The contour map preprocessor is much more complex, and was borrowed from the NPS Computer Center's library contour mapping routine CONTUR. The routine was modified so that instead of generating plot calls directly it now writes X,Y line coordinates to disk file FT03F001 for future reading by the PLOT program. The modified program is available from the author.

III. Code Reconciliation Program - RECON

A. Introduction

In a major programming effort (such as the STAR model) where several people are independently developing program modules to be added to the model, the problem of bringing together the developing modules into a common master code is time consuming and liable to oversight errors. A code reconciliation program named RECON has been written to simplify this task. It presumes that two programs (call them A and B) have evolved from some common ancestor code. A and B are thus mostly identical, but involve changes to some subroutines of the model, the addition of some new routines, and possibly also deletion of some routines. The code reconciliation program reads the source code for the A and B programs and prints out information about routines that match, lines of code which differ, and routines that are present in one of A or B but absent from the other.

The RECON program is written in SIMSCRIPT. It is a fairly simple program as word-processors go, but is too complex to describe in detail here. The source code is available from the author. In this report we concentrate on the features, use, and limitations of the RECON program.

B. Inputs

The main inputs to the RECON program are the two programs, A and B, to be compared. Program A is read from SIMSCRIPT input unit 1 and program B is read from unit 2. The user can define these units to point to whatever physical devices are desired - usually to disk data sets containing the source program card images.

For example, the following IBM/360 JCL points unit 1 to a program A stored in three source program modules on a disk. The three modules are automatically concatenated in the order given.

```
//GO.SIMU01  DD  UNIT=2314,VOL=SER=DISK03,DISP=SHR,DSN=PROGA1
//           DD  UNIT=2314,VOL=SER=DISK03,DISP=SHR,DSN=PROGA2
//           DD  UNIT=2314,VOL=SER=DISK03,DISP=SHR,DSN=PROGA3
```

In addition, there are 3 inputs which are read from unit 5 -- usually 3 cards. The first card contains an integer input value FILE.LGTH in free format. It defines the maximum number of card images from each program kept in RECON arrays at any given time. If RECON finds a mismatch in a given routine of the A and B programs, then it will look up to FILE.LGTH cards ahead in each routine trying to reestablish a match. A reasonable value for FILE.LGTH is 50, larger values may slow execution since up to FILE.LGTH**2 card comparisons may be made for each failure to match.

The second input card contains an alphabetic description of program A (for example, its name) which is printed on the RECON output for identification purposes. Up to 80 characters may be used. The third input card describes program B in an analogous fashion.

C. Module Definition and Sequencing

RECON breaks each of programs A and B into a sequence of modules. A module is a segment of code which starts with a header card and ends with the word END.

Header cards are cards on which the first non-blank characters are the keywords:

PREAMBLE
MAIN
EVENT
UPON
ROUTINE

corresponding to the standard SIMSCRIPT program elements. Each keyword must be followed by a blank(if there is room on the card for it). The EVENT, UPON, and ROUTINE header cards should also contain a program element name following the keyword. RECON will store up to 19-character names. A name must be followed by a blank column or by a left parenthesis if there is room for it on the card. The SIMSCRIPT keywords TO and FOR may appear between the header keyword and the name. Since RECON ignores TO and FOR in this context, these two words may not be used as module names.

The end of a module is signalled by any card that has the characters END separate from other characters(that is, immediately preceeded and followed by blanks if there is room on the card) before any '' comment indicators. END need not be the first word on the card.

RECON operates on one module at a time (in each of A and B). If the current module headers match (i.e. both PREAMBLE, both MAIN, both ROUTINE with matching name, or both EVENT or UPON with matching name) then RECON will go inside the module and compare its contents line by line. (See section D.). If the current module headers do not match, then RECON will print one of the modules as, for example,

"MODULE IN FILE A - NOT IN FILE B",

and advance to the next module header in that file to attempt a match again.

In order to determine which program file to advance if a match does not occur, RECON assumes that both programs contain modules in the following standardized order. First comes the PREAMBLE, followed by the MAIN program, and then all other EVENT, UPON, and ROUTINE modules arranged so that their names are in alphabetic order.

CAUTION: Alphabetic order is defined by the IBM SIMSCRIPT collating sequence which is

ABC → XYZ012 → 89 space dot

Note that the space (blank) comes after the alphanumeric characters but before the dot (period). This results in a slightly nonstandard alphabetization in which the following is a proper sequence of program element names:

RESET
RES2
RES
RES.MOVE

RECON uses the alphabetic order of names only when deciding which file to advance if headers do not match. If both A and B have the same module headers and names in the same sequence, then the alphabetic ordering will never be considered. But if A has modules named (in sequence) J, K, L, M, N while those of B are J, N, K, L, M, then RECON will

compare J's
declare module K in file A - not B
declare module L in file A - not B
declare module M in file A - not B

compare N's
declare module K in file B - not A
declare module L in file B - not A
declare module M in file B - not A

with the result being that many of the modules are not compared.

D. Comparisons Inside a Module

When RECON has located a pair of matching headers, it proceeds to do line by line comparisons of program A to program B within the module. It should be noted that only the header keyword and the name are used in matching header cards. Other information such as given and yielding arguments may be different on the header cards.

Lines from A and B including the headers are then compared character by character. If all 80 characters match, the lines are not printed, but a message is printed indicating the match:

"Matching block of 5 lines, A lines 16 to 20. B lines 17 to 21"

If the entire modules match, there will be only one such message for each module.

If any character of the current line in A fails to match the corresponding character of the current line in B, then the lines do not match. In this case RECON looks ahead in both files trying to find the closest place where a match can be reestablished. Up to FILE.LGTH future lines are considered in each of A and B (but never beyond the END of the current module). Messages identifying the non-matching portions are printed such as:

```
BLOCK IN FILE A -- NOT IN FILE B
     2 LINES, FROM LINE 24 TO LINE 25
24   LET  X = 1
25   CALL TOTAL
```

All non-matching lines are printed for easy cross reference to programs A and B.

If a match cannot be reestablished by the end of the modules or within FILE.LGTH lines, then the module is abandonned - remaining lines are not compared, and both programs are advanced to the next module.

Any program lines which do not lie between header and end cards are identified as being orphans and are printed.

IV. Code Plot Program

As an aid for documentation of computer programs, a code plotting program has been written. Computer listings of computer programs are often too faint to reproduce well in program documentation and may require photo-reduction to conveniently fit technical report or thesis page sizes. The code plotting program uses the VERSATEC plotter to produce crisp easily reproduced page size copies of card decks. Line numbers may also be added. This program is written in FORTRAN and calls standard VERSATEC graphics subroutines [1]. A program listing (plotted by the program itself) appears in Figure 3 and includes JCL and sample data cards.

The code plot program assumes the following input: the information to be plotted is read from cards -- one card per plot line. All 80 card columns are plotted. The input deck of cards to be plotted is arranged into modules each of which starts a new plot page. Each module starts with a line number card (which is not plotted) and ends with a nines card (which is also not plotted.)

The line number card contains a single input integer N in I5 format. If $N \geq 0$, line numbers will be plotted at the left of each line of information. The first line number is N and successive line numbers increase by one for each line. If N = -1, the line number sequence continues uninterrupted from the previous module. If N = -2, no line numbers will be plotted.

The nines card signals the end of a module and terminates plotting for the current page. It contains 9's in the first eight column positions. (Thus it is impossible to plot a card which starts with 8 nines.)

Up to 50 lines are plotted on each page. Modules which are longer than 50 lines automatically continue on additional pages with the line number sequence unbroken until a nines card is encountered.

```
STANDARD JOB CARD WITH TIME=1 GOES HERE
// EXEC FORTCLGW
//FORT.SYSIN DD *
C PLOT PROGRAM FOR 80 CHAR CARD IMAGES IN TECH REPORT FORMAT
      DIMENSION XBOX(7), YBOX(7), ITEXT(20)
      DATA XBOX/0.0,0.0,8.5,8.5,0.0,0.0,1.0/,
     -     YBOX/0.0,11.0,11.0,0.0,0.0,0.0,1.0/,  ICHK/'9999'/
      CALL PLOTS(0,0,0)
      XST = 0.0
      ZLINE = 1.0
C MAIN LOOP START NEW CODE SEGMENT
C GET LINE NUMBER CODE
50    READ (5,127,END=300) N
127   FORMAT(I5)
      IF (N.GE.0) ZLINE = N
C     MOVE OVER FOR NEW PAGE
100   XST = XST +  9.0
      DO 120 I = 1,5
120   XBOX(I) = XBOX(I) + 9.0
C OUTLINE THE PAGE
      CALL NEWPEN(1)
      CALL LINE(XBOX,YBOX,5,1,0,0)
C PRINT THE PAGE
      CALL NEWPEN(2)
      X = XST + 2.0
      XX = XST + 1.5
      Y = 9.5
      DO 200 I = 1,50
      READ(5,157,END=300) ITEXT
157   FORMAT(20A4)
      IF((ITEXT(1).EQ.ICHK).AND.(ITEXT(2).EQ.ICHK)) GO TO 50
      Y = Y - 0.15
      CALL SYMBOL(X,Y,0.070,ITEXT,0.0,80)
      IF (N .EQ. -2) GO TO 200
      CALL NUMBER(XX,Y,0.07,ZLINE,0.0,-1)
      ZLINE = ZLINE + 1
200   CONTINUE
      GO TO 100
300   CALL PLOT (0.,0.,999)
      STOP
      END
REPLACE THIS CARD WITH A STANDARD ORANGE END OF FILE CARD
//GO.PLOTPARM DD *
 &PLOT  XMIN=6.0,XMAX=150.0,YMIN=-3.0,YMAX=12.0  &END
//GO.SYSIN DD DATA
LINE NUMBER CARD GOES HERE
INPUT DECK TO BE PLOTTED GOES HERE
NINES CARD GOES HERE
REPLACE THIS CARD WITH A STANDARD ORANGE END OF FILE CARD
```

Figure 3. Code Plot Program

The last module in the input deck is ended with a nines card and followed by the standard orange end of file card (/*).

REFERENCES

[1] "VERSATEC GRAPHICS PLOTTING MANUAL", Naval Postgraduate School, W.R. Church Computer Center, Technical Note No. 0141-34, Feb. 1978.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 55 Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
4.	Professor James K. Hartman Code 55Hh Department of Operations Research Naval Postgraduate School Monterey, California 93940	40
5.	Professor S. H. Parry, Code 55Py Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
6.	LTC Edward P. Kelleher, Code 55Ka Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
7.	Professor Arthur L. Schoenstadt, Code 53Zh Department of Mathematics Naval Postgraduate School Monterey, California 93940	1
8.	Office of the Commanding General U.S. Army TRADOC Attn: General Donn A. Starry Ft. Monroe, Virginia 23651	1
9.	Headquarters U.S. Army Training & Doctrine Command Attn: ATCG-T (Col. Ed Scribner) Ft. Monroe, Virginia 23651	1

10. Headquarters | 1
U.S. Army Training & Doctrine Command
Attn: Director, Analysis Directorate
Combat Developments (MAJ Chris Needels)
Ft. Monroe, Virginia 23651

11. R. Stampfel, Code 55 | 1
Department of Operations Research
Naval Postgraduate School
Monterey, California 93940

END

DATE
FILMED
8-80

DTIC